branches to step **1112** wherein the replacement is made, otherwise there is nothing to replace and step **1110** bypasses any replacement. In any event, the process continues to step **1114**.

[0099] At this time, the appropriate assembly version is known, as specified in the manifest and as altered via any configuration instructions, as described above. Step **1114** enumerates any dependencies in the assembly manifest that corresponds to this appropriate assembly for handling in a similar manner. These and other identified assemblies may be handled in a similar manner so that the correct versions as specified in the configurations are bound to the application.

[0100] As can be seen from the foregoing detailed description, there is provided a method, system and infrastructure to version globally named objects in the system. Application authors may create safe, isolated applications by simply creating a declarative manifest that describes dependencies on these shared objects, without needing to be concerned with coding the application to adjust to the version of the assembly being used. By the present invention, assemblies can be safely shared and applications can be more completely isolated.

[0101] While the invention is susceptible to various modifications and alternative constructions, certain illustrated embodiments thereof are shown in the drawings and have been described above in detail. It should be understood, however, that there is no intention to limit the invention to the specific form or forms disclosed, but on the contrary, the intention is to cover all modifications, alternative constructions, and equivalents falling within the spirit and scope of the invention.

What is claimed is:

1. A computer-implemented method, comprising:

  receiving a request corresponding to binding at least one shared assembly to executable code; and

  interpreting configuration information to determine a version of a shared assembly to bind to the executable code, wherein the configuration information is separate from the shared assembly.

2. The computer-implemented method of claim 1 wherein interpreting configuration information includes redirecting one assembly version to another assembly version.

3. The computer-implemented method of claim 1 wherein the executable code comprises an application program, and further comprising, associating the application program with an application configuration having redirection information therein.

4. The computer-implemented method of claim 3 wherein associating the application configuration with the application program comprises storing the application configuration in a folder containing the application program.

5. The computer-implemented method of claim 3 wherein interpreting configuration information includes redirecting an assembly version from the other assembly version to a third assembly version according to at least one other configuration.

6. The computer-implemented method of claim 5 wherein the at least one other configuration comprises a publisher configuration.

7. The computer-implemented method of claim 5 wherein the at least one other configuration comprises an administrator configuration.

8. The computer-implemented method of claim 1 further comprising, associating at least one assembly version with a publisher configuration having redirection information therein, wherein interpreting configuration information includes interpreting the publisher configuration.

9. The computer-implemented method of claim 8 wherein associating the publisher configuration with at least one of the assembly versions comprises storing the publisher configuration in an assembly cache containing at least one of the assembly versions.

10. The computer-implemented method of claim 1 wherein interpreting configuration information includes interpreting an administrator configuration.

11. The computer-implemented method of claim 10 further comprising, storing the administrator configuration in a system folder.

12. The computer-implemented method of claim 1 further comprising, caching data identifying the version of the shared assembly determined from interpreting the configuration information.

13. The computer-implemented method of claim 1 wherein the executable code comprises an application program, and wherein interpreting configuration information includes interpreting at least one of an application configuration, a publisher configuration and an administrator configuration, each configuration having redirection information therein for redirecting one assembly version to another assembly version.

14. The computer-implemented method of claim 1 wherein the executable code comprises an application program, and wherein interpreting configuration information includes interpreting at least one of a publisher configuration and an application configuration, each configuration having redirection information therein for redirecting one assembly version to another assembly version.

15. The computer-implemented method of claim 1 wherein the executable code comprises an application program, and wherein the configuration comprises an application configuration, a publisher configuration and an administrator configuration, and wherein interpreting the configuration includes, determining whether a safe mode of operation is present, and if so, interpreting the application configuration and the administrator configuration but not interpreting the publisher configuration.

16. A computer-readable medium having computer-executable instructions for performing the method of claim 1.

17. A system in a computing environment, comprising:

  an manifest including information that specifies a dependency of executable code on an identified version of a shared assembly;

  a configuration including information that redirects at least one version of a shared assembly to another version of that shared assembly; and

  a binding mechanism configured to receive a request directed to execute the executable code, to select the identified version of the shared assembly from the manifest, and to interpret the configuration to determine whether to redirect the identified version in the manifest to another version identified in the configuration.